

Projet : Description d'un contrôleur de SRAM ZBT

L'objectif de ce projet est de décrire un contrôleur de mémoire SRAM. Le contrôleur devra permettre de piloter une mémoire CYPRESS mt551512y36f. Vous pourrez vous appuyer sur la documentation de cette mémoire et vous utiliserez son modèle VHDL (mt551512y36f.vhd) fournie par le fabricant. Ce type de mémoire statique porte l'extension ZBT, cela signifie qu'il n'y a pas de perte de cycle entre une phase de lecture et d'écriture (Cf figure 1).

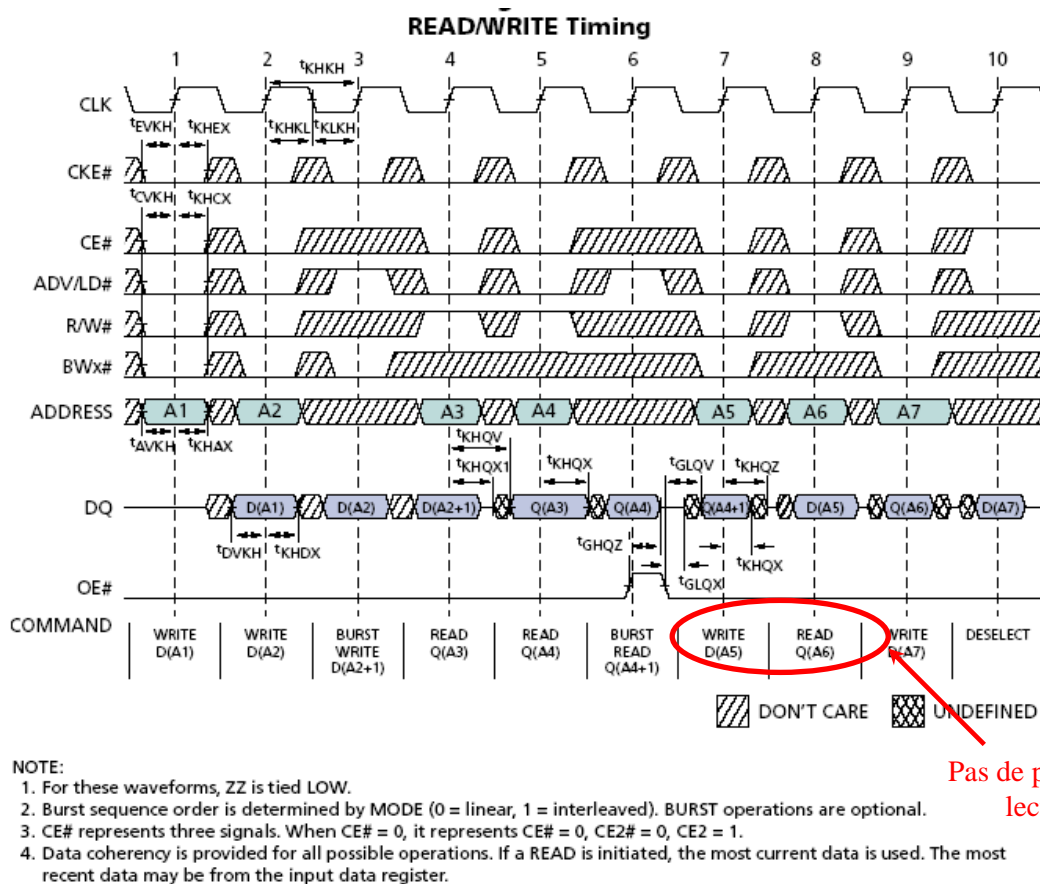
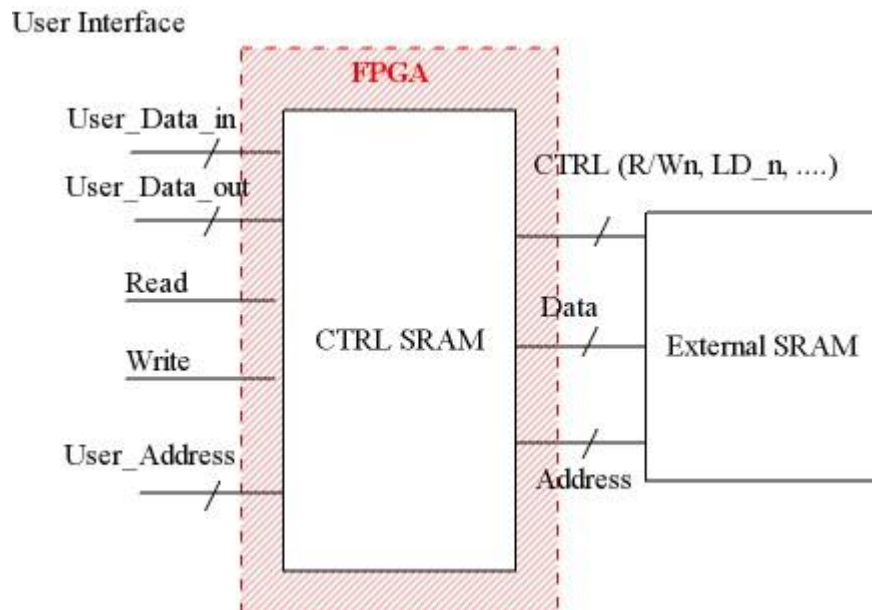


Figure 1

Vous devrez gérer les signaux de contrôle CKE#, CE#, OE#, R/W#, (ADV/LD# = LD_n), CE2#, CE2, les bus d'adresses et de données (appuyer fortement sur le diagramme de la figure 1). Attention le bus de données est bidirectionnel, il devra donc être géré pour cela avec des buffers 3 états (composant IOBUF cf Annexe 1).

Travail à réaliser :

Vous modéliserez le contrôleur **en C puis en VHDL** sans vous préoccuper des modes BURST. Le modèle en langage C doit être transformé en VHDL via une approche HLS pour être comparé en terme de performance au modèle décrit manuellement. Vous effectuerez le testbench vhd associé. La génération d'adresse sera effectuée à l'extérieur du composant. Le contrôleur possédera en outre obligatoirement deux signaux d'entrée READ et WRITE. Ces signaux déclancheront respectivement des phases de lecture ou d'écriture dans la mémoire SRAM. Le signal READ sera prioritaire sur le signal WRITE.



Question subsidiaire : Modifier le composant pour réaliser une suite d'écriture ou de lecture à partir d'une adresse de départ.

Conseil rendre les signaux suivant constants de la manière suivante

nBWA = '0'
 nBWB = '0'
 nBWC = '0'
 nBWD = '0'
 ZZ = '0'
 MODE = Lbo_n = '0'
 nCKE = '0'

Annexe 1 :

Pour la description VHDL utiliser des primitives Xilinx, utiliser la librairie unisim

```
library IEEE;
use IEEE.std_logic_1164.all;
-- synopsys translate_off
    library unisim;
    use unisim.vcomponents.all;
-- synopsys translate_on
```

Utiliser les composants IOBUF presents dans la librairie unisim pour utiliser des buffers tristate

```
component IOBUF
    port (I : in std_logic;
          IO: inout std_logic;
          T : in std_logic;
          O : out std_logic);
end component;
```